SELECTIVE CALLING DEVICE FOR THE MOBILE MARITIME SERVICE

Vernon Christopher Hipkiss

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

Selective Calling Device
for
The Mobile Maritime Service

by

Vernon Christopher Hipkiss

Thesis Advisor: V. M. Powers

September 1973

Selective Calling Device for The Mobile Maritime Service

by

Vernon Christopher Hipkiss
Lieutenant, United States Coast Guard
B.S., United States Coast Guard Academy, 1966

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 1973

ABSTRACT

INTEL's MCS-4 micro computer set was selected to implement a digital selective calling device for use by the mobile maritime service. The system will generate and receive a standardized calling message. Upon receipt of a valid calling message, intended for the user, the system will activate visual and audible alarms.

TABLE OF CONTENTS

I. INTRODUCTION

The need for a world wide selective calling system for the mobile maritime service has been long recognized and it appears that the International Telecommunications Union (ITU) will make a final determination as to the type and form in the coming year. The following material presents a particular approach to the development of such a system based on a message form developed by the United States.

In brief this selective calling system will generate and decode a standardized message which can be used for ship to ship or ship to shore communications. The equipment will be modular in design and will interface readily with existing communications equipment. The system will generate and respond to four different bit rates and several address forms. It will also have the versatility to be expanded to provide more sophisticated users with various degrees of flexibility.

The selective calling system presented here will provide a basic system that will compose a calling message from a keyboard input and will store the message in a random access memory. Then, upon a signal from the keyboard operator, it will transmit the message at a selected bit-rate. The calling message will remain in memory to allow for retransmission if desired. The system will also receive an incoming message, store it in memory, and deter-

mine whether or not the message is intended for the particular user. If it is for the user the system will energize two indicator lights, one for the priority of the call and the other for the type of the call.

In developing this system the INTEL CORP. MCS-4 Micro Computer Set was chosen to provide the nucleus of the selective calling unit. It is felt that although the MCS-4 is a 4-bit machine, it is capable of providing the necessary service. It was also desired to use "off the shelf" equipment wherever possible.

The following material presents the background behind the selective calling system, the design of, and the realization of one possible form of the system. Chapter II contains information on the background development of the selective calling system. Chapter III contains the basic description of the calling system being presented in this paper. Chapter IV describes how the system will be implemented and gives the basic programing that will be used. Chapter V goes into the program with more detail, showing a flow diagram and a detailed description of the function blocks. Chapter VI gives some conclusions, while Appendix A contains the calling message format. Appendix B contains a listing of the MCS-4 instructions used in the program and Appendix C contains the complete program with notations.

II. BACKGROUND

A communications link is established between a ship and a shore station on frequencies allocated by the International Telecommunications Union (ITU). Three modes of communications are involved in the maritime service: MF/HF radiotelegraphy, MF/HF radiotelephony, and VHF radiotelephony.

To establish a communications link between a ship and a shore station certain calling procedures, as defined by the radio regulations, are carried out. In brief the calling station will come up on a calling frequency and contact the called station by call sign. The called station must be listening for the call. Upon establishing contact, both stations then shift to a clear working frequency. In order to maintain twenty-four hour communications both stations must maintain constant watches. This is done in the military service and on large passenger vessels but is not practical for the normal merchant vessel. Coast stations do maintain a twenty-four hour guard because their traffic load warrents it. Therefore, it is normally much easier to establish ship-to-shore communications then it is to establish shore-to-ship communications.

Contact potential, under these conditions, is poor with most merchant ships carrying only one radio operator. A one-operator schedule consists of two four-hour

watches per day; the schedule is documented so that contact procedures are not entirely hit and miss. What is hit and miss is that one operator can only guard so many frequencies at a single time, and with the signal propagation path changing throughout the day more delays are encountered. Because of these situations, it is generally estimated that the average contact time runs about six hours. The possibility of reducing this time under present procedures is very unlikely.

The U.S. merchant fleet is being modernized with new and larger ships in order to compete with foreign fleets. The new ships are highly automated and, in order to be competitive, must be utilized to their utmost capabilities. This in turn requires the highest standards of management from the shore base owners. To attempt to maintain the required efficiency with communications links as weak as before mentioned is not consistent with modern techniques.

In the area of search and rescue procedures, delayed contact times can mean the difference between life and death. The development of the AMVER system (Alantic Merchant Vessel Reporting) has given the rescuer a means to pinpoint, in a matter of seconds, the estimated position of most ships traveling the world's seaways. It is therefore possibile for a ship to be in an area of a distress and be unaware of the situation because of a lack of communications. The above situation is partially covered thru the use of the AUTO ALARM system. This system con-

sists of the transmission of a series of CW pulses that causes unattended decoder equipment to become activated; however when this system is energized, all ships within the broadcast range of the transmitter are alerted, even those who would be unable to assist. It can readily be seen that AMVER coupled with the ability to selectively contact any ship at any hour will enhance present search and rescue procedures.

These problems have long been recognized internationally and in 1959 the C.C.I.R. (International Radio Consultative Committee) described a selective calling system. In 1963 and 1965 study groups, set up by the C.C.I.R., concluded that a world wide selective calling system should be standardized as soon as possible.

In 1972 the United States proposed a digital, narrow banded selective calling system. GTE Sylvania was given a contract to implement and test this system for the United States Maritime Administration. In January of next year the final meetings of the study groups will be held to discuss selective calling systems. The operational tests conducted by the United States will be presented at this time and, if accepted by the C.C.I.R., the system will then be presented to the Maritime Administrative Radio Conference. If accepted, the system will be adopted into the Radio Regulations of the I.T.U.

III. SYSTEM DESCRIPTION

The automatic reception and decoding of a calling message is the primary purpose of the selective calling system. The individual units will be designed so as to operate in both the transmit and receive modes and will be readily compatible with existing communications equipment. The individual units will be modular in design so that add-on features will be compatible with the basic unit. The basic units will be capable of address detection and indication and message priority indication, and will be able to generate a calling message.

The selective calling system is based on a standardized message consisting of a synchronization portion followed by an informational portion. One character, refered to as the 'mode', designates the priority of the message (Distress, Urgency, Safety). Two other characters are refered to as 'lable 1' and 'lable 2'. Lable 1 designates the type of message (Selective Call, All Ships Call, Group Call, or Geographic Area Call). Lable 2 is provided to allow a means of turning on, or off, peripheral equipment. The rest of the message is devoted to the call signs of the calling and called parties, the radio frequency over which continued communications is desired and a 'text' portion.

The system being developed will utilize the Interna-

tional Telegraph Alphabet No. 5 (ITA-5) which is equivalent to ASCII code. This seven bit code will be augmented by an eighth parity bit with the sense of the parity being odd over the eight bits. Frequency Shift Keying (FSK) will be used to transmit all calling messages as follows:

1. VHF radiotelephony: 1200 or 600 bits per second with a frequency shift of ±425 Hz about 1900 Hz.
2. MF/HF radiotelephony: 150 or 75 bits per second with a frequency shift of ±85 Hz about 2500 Hz.
3. MF/HF radiotelegraphy: 150 or 75 bits per second with a frequency shift of ±85 Hz about 2500 Hz.

The major parts of the calling message are the seven character call sign of the called party which is repeated within the message, the seven digit reply frequency, the seven character call sign of the calling party, and a text portion. The call sign of the called party is repeated to help detect errors and prevent false alarms. The synchronization portion of the message contains sixteen '*'s, two 'SYN' characters and an 'SOH' character. The 'SOH' character preceeds the informational portion of the message.

The selective calling system, with its unit modular design, can provide optional capabilities, such as an alpha-numeric display. The display would be used as an aid in the composition of outgoing messages, and used to display the major portions of the incoming messages. These portions would be the call sign of the calling sta-

tion, the frequency for reply, and the text portion of the message. An additional option would be a 'lable 2' decoder. This decoder would activate a signal to turn on, or off, peripheral equipment such as a teletype, facsimile, or audio tape recorder.

A detailed description of the calling message format is contained in Appendix A.

IV. IMPLEMENTATION

INTEL's MCS-4 micro computer set was selected to implement the selective calling system. The set consists of a 4-bit parallel CPU with the capability of performing 45 instructions. The CPU can drive from one to sixteen ROM's and from one to sixteen RAM's. A minimum configuration would consist of one CPU chip and one ROM chip. The CPU chip contains sixteen 4-bit general purpose registers and the capability of nesting up to three levels of subroutines. The 2048 bit ROM's are metal mask programable with each chip being organized into 256 eight bit words. The 320 bit RAM's are arranged into four registers with each register containing twenty 4-bit words. In addition, each ROM and RAM has associated with it four external input/output (I/O) lines. INTEL also provides a 10-bit serial-in/parallel-out, serial-out shift register that can be added to the basic computer set.

The MCS-4 was selected because of its availability and low cost. This in no way excludes other micro computers from being used in a similar implementation. A more detailed description of the MCS-4 can be obtained from INTEL, and Appendix B contains a listing of the MCS-4 instructions used in this paper.

In designing the selective calling unit around the MCS-4 micro computer set, it was decided to use the MCS-4

for all functions of the system which it could implement. That is, the only other associated equipment would be the interfaces between the computer and the keyboard, FSK coders/decoders, and the display lights.

The goal of the design was to develope a unit that would generate and transmit, receive and decode, a calling message at anyone of four different transmission speeds. The calling message could be anyone of the four proposed types: Selective (individual user), All (all users), Group (group of users), Area (any user in a specified geographic area). In the latter case, the user would be required to to change his area call as he traversed specific geographic ares. The display of the unit would consist of seven lights. Three of the lights would be associated with the priority of the call and the other four would indicate the type of call. An audible alarm would be provided when any of the indicator lights are lit.

Since the timing of the outgoing message and the synchronization of the incoming bit stream would be provided by the MCS-4, timing was the most critical design consideration. The MCS-4 has an instruction cycle of 10.8 usec; however this instruction cycle speed does not relate exactly to the specified message bit rates. At 1200 bits per second, approximately 77 instructions would be required between the reading of each bit. In order to achieve exactly 77 instructions, the instruction cycle must be changed to 10.823 usec. The MCS-4 uses an external timing

source which provides eight pulses for every instruction cycle, thus an oscillator with a frequency of 739.20±.2 KHz would provide a properly timed instruction. Using 77 instructions as a timing cycle for 1200 bits per second, two such cycles would correspond to 600 bits per second, eight cycles for 150 and sixteen cycles for 75 bits per second.

When an incoming bit stream is sensed by the MCS-4, the bit stream is sampled at periods of one timing cycle. The bit stream corresponding to the string of '*'s that precedes every calling message will appear as 0101010001010... Now, by counting the number of cycles for a '1' and the number of cycles for a '0' and comparing them, a bit rate can be determined. A complete description of the mechanics of this determination will be covered in the next chapter, which descibes the actual program.

After establishing the bit rate, the character 'SOH', which has a bit configuration 10000000, is searched for. In this case, the computer, using the bit rate already established, looks for seven '0's in a row. This not only indicates the begining of the main portion of the message but also allows the computer to align the bit stream for proper dissection into eight-bit characters. It should be noted that the calling message consists of a continous information bit stream with no idle or start bits, as is found in standard TTY transmissions.

Once the start of the message is established and the proper alignment is attained, the bit stream is broken

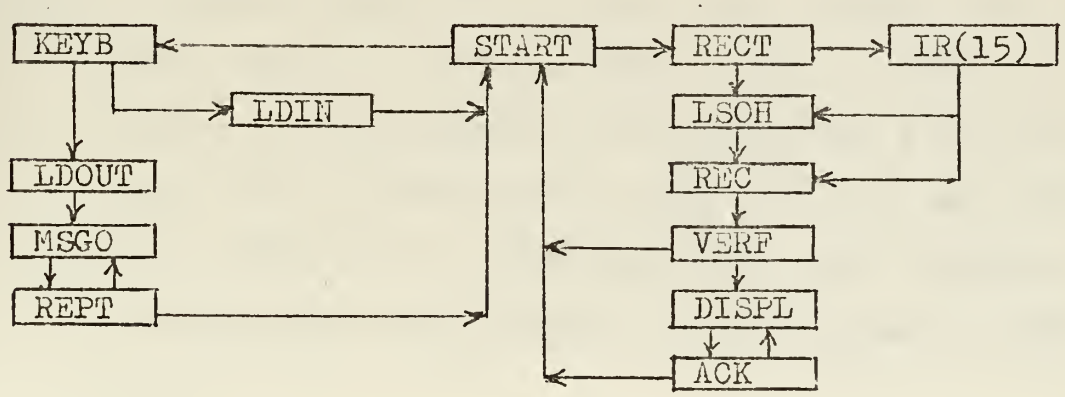
up into 4-bit bytes and stored in the RAM's for further examination. (Throughout the rest of this paper a 4-bit group will be refered to as a byte and two bytes, or eight bits, will be called a character). After the message is loaded into the RAM'S, it is a simple matter to compare the sent call sign with call signs that have been preloaded within the computer's memory bank. If the appropriate call signs are the same, then the associated display lights are energized. If the call signs do not match, the system returns to the start of the program and looks for another incoming message.

When the operator desires to transmit a calling message, he alerts the computer to this fact by a signal from the keyboard. This switches the computer from a receive condition to a keyboard condition. The operator will then, through an input from the keyboard, signal the computer to prepare for a 'loadout', or a 'loadin', operation. (Loadout would be loading a calling message for transmission. Loadin would be loading the users own call signs for Selective, Group, Area and All Ships). If it is a 'loadin', the computer will accept 32 characters from the keyboard and then return to the start condition. If it is a 'loadout', the computer will accept 48 characters from the keyboard, corresponding to the informational portion of the calling message, and store them in RAM's. The sixteen '*'s will be loaded automatically. The entire calling message is sent after the computer receives an input from

the keyboard. Depending on the next input from the keyboard, either the calling message will be retransmitted, or the system will return to the start condition waiting for incoming signals.

V. PROGRAM

The following material is broken down into a flow diagram of the computer program and a detailed description of each function block. Each MCS-4 input or output port is a group of four lines. All input lines to the computer are connected to ROM ports 0, 1, and 2. All output lines are connected to RAM ports 0, 1, and 2. The keyboard is connected to ROM ports 1 and 2, while the input FSK decoder is connected to bit position 3 of ROM port 0. Bit position 0 of RAM port 0 is connected to the FSK output encoder, while bit positions 0 thru 2 of RAM ports 1 and 2 are connected to the display indicators. Bit position 3 of RAM port 1 provides the reset signal to the keyboard after a character has been entered. The test pin of the MCS-4's CPU is connected to the keyboard to signal when a key has been depressed. Following is a functional block diagram of the selective calling program.



The following is a short description of each function block.

START: Await incoming message or keyboard signal.

RECT: Receive timing determination.

LSOH: Locate 'SOH'.

REC: Receive message and store in RAM's.

VERF: Verify incoming call sign against preset call signs.

DISPL: Display mode and lable 1.

ACK: Acknowledge the receipt of the display, then jump to START.

KEYB: Receive keyboard signal then jump to LDIN or LDOUT.

LDIN: Load in reference call signs.

LDOUT: Load calling message for transmission.

MSGO: Transmit calling message at a preset bit rate.

REPT: Retransmit message or jump to START.

IR(15): Store code to determine bit rate.

Following is a detailed description of each function block with reference to appendix C where the complete program will be found.

START: Instructions 0000 to 000F form a closed loop with exit points to RECT and KEYB. Exit to RECT occurs when a '1' is detected at bit position 3 of ROM 0, while exit to KEYB occurs when four '1's are detected at ROM port 1. Depressing the 'DEL' character on the keyboard would generate such a signal at ROM port 1.

RECT: Instructions 0082 to 00C5 examine the incoming bit

stream of '*'s which will appear in time sequence at bit position 3 of ROM port 0 as:

```
0101010001010100010101000l.....
ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

The letters will be used in the discussion below as a reference to identify individual bits. When a '1' is detected at the read port, IR(15) is incremented and one timing cycle (77 instructions) is conducted and the port is read again. If it is still a '1', IR(15) is again incremented and another timing cycle conducted. This routine is continued until a '0' is detected at the port. IR(15) now contains the number of timing cycles utilized in reading a '1'. The routine now continues, repeating a similar process, now looking for a '1' to indicate the end of the '0' pulse. The count of the timing cycles utilized is kept in IR(14). The routine now compares IR(15) and IR(14) to see if they are equal and if they are, indicating equal length pulses, the routine returns to timing the length of the '1' and '0' pulses. If IR(14) and IR(15) are again found to be equal, a bit stream 10101 will have been determined. This would correspond to bits B thru F as shown above. Now the routine, using the timing established by the count in IR(15), waits out four pulse lengths (F,G,H and I) and starts again to read '1's at bit position J. If the 10101 pattern is received for three more consecutive times, the bit timing and the fact that it

is a string of '*'s are established. If at any time IR(15) and IR(14) are not equal, indicating uneven pulses such as 110 etc., the routine waits for a '0' and then returns to START waiting for a '1'. This prevents the routine from getting into an endless loop within RECT.

LSOH: Instructions 00C7 to 00D7 look for the character 'SOH' which has a bit pattern 10000000 with the '1' being the LSB. The routine looks for the seven '0's in a row to indicate the last seven bits of the 'SOH' and thus the begining of the informational part of the calling message.

REC: Instructions 00D9 to 010B are a routine that takes one bit at a time, received at bit position 3 of ROM port 0, and arranges them into bytes. The bytes are then stored in RAM's "0" and "1", filling six registers with 48 successive characters (96 bytes).

VERF: Instructions 011D to 015D are a routine that compares the repeated call sign to determine if it is a valid calling message and then compares the call sign of the message against the preset call signs stored in RAM 2. If the message is intended for this particular unit, the routine accepts the call by jumping to the display portion of the program. If the message is not accepted, the routine jumps to START.

DISPL: Instructions 015F to 016B are a routine that dis-

plays the mode and lable 1 character at bit position 0 thru 2 of RAM ports "1" and "2" respectively. The mode character can be either A, B or C with the three LSBs of each being 001, 010 or 011 respectively. The lable 1 character can be either A, B, C or D with the three LSBs being 001, 010, 011 or 100 respectively. These three bits are used to light the displays.

ACK: Instructions 016D to 0171 are a routine that waits for an input from the keyboard (any input) to send the program back to START.

KEYB: Instructions 0011 to 0026 wait for an input from the keyboard at ROM port 1. The program jumps to LDIN if the port receives four '1's (1111) and will jump to LDOUT if the port receives (1110) with the '0' being the LSB. One set of keyboard characters that corresponds to these inputs are: 'DEL'=(1111) and 'N'=(1110). If the signal from the keyboard is other than the above bit patterns the routine jumps to the start of the routine and waits for another input.

LDIN: Instructions 0028 to 0038 are a routine that loads RAM 2, registers 0 thru 3 with inputs from the keyboard. Register 0 holds the user's "selective" call sign, register 1 holds the "all ships" call, register 2 holds the "group" call sign and register 3 holds the "area" call sign. Once the four registers are filled the program jumps back to START. The

LDIN routine, as well as the LDOUT routine, calls on the subroutine TNFR. TNFR then transfers a keyboard character, as two bytes, to a preselected RAM register. In the LDIN routine, eight characters must be entered from the keyboard for each seven character call sign. The eighth character can be a random character, as it does not affect the program.

LDOUT: Instructions 003A to 0058 are a routine that loads RAM 0 and RAM 1 with keyboard inputs similar to the LDIN routine. Specifically, registers 2 and 3 of RAM 0, and registers 0 thru 3 of RAM 1, are loaded from the keyboard. This corresponds to the informational portion of the calling message starting with the character 'SYN' and ending with 'ETX'. One additional character must be entered to fully fill the final register. (This is necessary because of the TNFR subroutine) The sixteen '*'s are automatically loaded by the LDOUT routine into registers 0 and 1 of RAM 0.

MSGO: Instructions 005A to 006A are a routine that transmits the calling message. The first part of this routine loads from the keyboard, and stores into IR(15), the four bits that establish the transmission speed. That is, striking the keyboard character "1", "2", "8" or "0" establishes speeds of 1200, 600, 150 or 75 bits per second respectively. Following this input, the computer waits for any input from the

keyboard which initiates the transmission of the stored calling message held in the RAM's. The subroutine LD8 performs the necessary byte manipulation to provide a serial bit stream at bit position 0 of RAM port 0.

REPT: Instructions 006C to 008C are a routine that waits for an input from the keyboard to signal the program to return to START or to retransmit the calling message. If the 'DEL' keyboard character is struck the message is retransmitted. If the '0' keyboard character is struck the program returns to START.

VI. CONCLUSIONS

The program contained in Appendix C was simulated, using a FORTRAN program, on an IBM 360. The results, from the simulation, indicate the MCS-4 design would function as a selective calling system. The next step, leading to a full implementation of the system, would be to connect the micro computer set to the keyboard, display lights and FSK encoders/decoders. The MCS-4, using simple transistor interfacing, is readily compatible with these types of equipment. The system could then be bench tested and the results should be the same as those obtained through the simulation. It then remains to interface the units with communications equipment and test under actual conditions.

It still remains to be seen if under actual conditions, with signal interference, what sort of reliability would be attained. Another problem that might arise with the program is the timing. The oscillator must be accurate if proper bit alignment is to be maintained, as there is no practical means to realign once the bit stream enters the RAM's. A computer with some sort of interrupt that could realign everytime a '1' appeard in the bit stream would be more advantageous. The BCS check was not implemented in the program as this is best done with external hardware.

Even with these problems, this system has a great advantage over present methods of communications. It will

provide a quick means of contacting users when the radio operators are off watch. This will be useful both for normal operations and SAR (Search And Rescue) operations. The system, although more expensive than no system, will be relatively cheap. The system also has a very short start to stop sending time. This will allow more individual calls to be sent on the presently overloaded communications circuits. Even if this system is more vulnerable to interference, the probability of making contact would go up with repeated transmissions.

APPENDIX A

SELECTIVE CALLING MESSAGE-CHARACTER SEQUENCE

Character Number	Symbol/ Legend	Remarks and Instructions
0 to 15	*	Asterisk, repeated 16 times
16 to 17	'SYN'	Repeated 2 times
18	'SOH'	Start of Header
19		Mode Character. See Note 1
20 to 26		Called party call sign. See Note 2.
27	'ACK/NAK'	Acknowledge
28		Lable 1. See Note 3
29		Lable 2. See Note 3
30	'SP'	Space (non-printing)
31 to 37		Repeat of 20 to 26
38	'SP'	Space (non-printing)
39 to 45		Frequency of reply (0.1 of a KHz)
46	'SP'	Space (non-printing)
47 to 53		Calling party call sign
54	'STX'	Start of Text
55 to 61		Text
62	'ETX'	End of Text
63 to 64	BCS	Block check sequence: CCITT polynomial

Note 1: Mode Characters

Type of Call	Mode Character
Distress	A
Urgency	B
Safety	C

Note 2: Called Party Call Sign

Selective Call: Seven characters taken from "A" through "Z" and "0" through "9". Precede call sign by a slant(/) to fill all seven characters.

Example: "/AML947"

Area Call: Six characters preceded by the character 'SUB'.

Example: "'SUB'123456"

All Ships: The character 'L', repeated six times, preceded by the character 'SUB'.

Example: "'SUB'LLLLLL"

Group Call: Six characters preceded by the character 'GS'.

Example: "'GS'CGUARD"

Note 3: Lable Control Characters

Lable 1: Type of Call	Lable 1 Character
Selective	A
All Ships	B
Geographic Area	C
Group	D

Lable 2: Meaning	Lable 2
Turn on audio tape recoder	A
Turn off audio tape recoder	B
Turn on teleprinter	C
Turn off teleprinter	D
Turn on facsimile	E
Turn off facsimile	F

APPENDIX B

PARTIAL LIST OF MCS-4 INSTRUCTIONS

Mnemonic	OPR	OPA	Description of Operation
NOP	0000	0000	No operation
JCN	0001 -A2-	CCCC -A1-	Jump to ROM address -A2-, -A1- (within the same ROM that contains the JCN instruction) if condition CCCC is true, otherwise skip. See Notes 1 and 4
FIM	0010 -D2-	RRR0 -D1-	Fetch immediate from ROM date -D2-,-D1- to index register pair location RRR. See Note 2.
SRC	0010	RRR1	Send register control
JUN	0100 -A2-	-A3- -A1-	Jump unconditional to ROM address -A3-,-A2-,-A1-
JMS	0101 -A2-	-A3- -A1-	Jump to subroutine ROM address -A3-, -A2-,-A1- save old address
INC	0110	RRRR	Increment contents of register RRRR See Note 3
ISZ	0111 -A2-	RRRR -A1-	Increment contents of register RRRR, go to ROM address -A2-,-A1- if RRRR≠0, otherwise skip
ADD	1000	RRRR	Add contents of register RRRR to accumulator with carry
SUB	1001	RRRR	Subtract contents of register RRRR from accumulator with borrow.

LD	1010	RRRR	Load register RRRR to accumulator
XCH	1011	RRRR	Exchange contents of index register RRRR with accumulator
BBL	1100	DDDD	Branch back and load data DDDD in the accumulator
LDM	1101	DDDD	Load data DDDD in the accumulator
WRM	1110	0000	Write the contents of the accumulator into previously selected RAM main memory character
WMP	1110	0001	Write the contents of the accumulator into the previously selected RAM output port
SBM	1110	1000	Subtract the previously selected RAM main memory character from the accumulator with borrow
RDM	1110	1001	Read the previously selected RAM main memory character into the accumulator
RDR	1110	1010	Read the contents of the previously selected ROM input port into the accumulator
CLC	1111	0001	Clear carry
IAC	1111	0010	Increment accumulator
CMA	1111	0100	Complement accumulator
RAL	1111	0101	Rotate left(accumulator and carry)
RAR	1111	0110	Rotate right(accumulator and carry)

Note 1: Condition codes (CCCC=$C_1C_2C_3C_4$)

C_1=1 Invert jump condition

C_1=0 Not invert jump condition

C_2=1 Jump if accumulator is zero

C_3=1 Jump if carry/link is a one

C_4=1 Jump if test signal is a zero

Note 2: RRR is the address of 1 of 8 index register pairs in the CPU.

Note 3: RRRR is the address of 1 of 16 index registers in the CPU.

Note 4: The notation -A2- stands for a 4-bit byte.

APPENDIX C

MCS-4 PROGRAM

A: Hexadecimal ROM address

B: ROM contents (machine language)

A	B	Instruction		Comments
0000	2000	START:	FIM 0, 00H	
0002	2210		FIM 2, 10H	
0004	23		SRC 2	
0005	EA		RDR	;Read ROM port 1
0006	F4		CMA	
0007	1411		JCN 4, KEYB	;Jump to KEYB if ACC=0
0009	21		SRC 0	
000A	F1		CLC	
000B	EA		RDR	;Read ROM port 0
000C	F5		RAL	;Bit (3) goes into carry
000D	1282		JCN 2, RECT	;Jump to RECT if CY=1
000F	4000		JUN START	;Loop back to START
0011	2010	KEYB:	FIM 0, 10H	
0013	2220		FIM 2, 20H	
0015	5173	ERR:	JMS RSET	;Reset KBD for next input
0017	5173	WAIT:	SRC 0	
0018	1117		JCN 1, WAIT	;No KBD character; wait
001A	00	SWA:	NOP	;Short wait loop to insure
001B	711A		ISZ 1, SWA	;KBD contacts closed
001D	EA		RDR	;Read ROM port 1
001E	F4		CMA	
001F	1428		JCN 4, LDIN	;Reference call signs, jump ;LDIN if ACC=0
0021	EA		RDR	Read ROM port 1
0022	F2		IAC	
0023	F4		CMA	
0024	143A		JCN 4, LDOUT	;Outgoing message, jump to ;LDOUT if ACC=0
0026	4015		JUN ERR	;Wrong input loop to ERR
0028	2480	LDIN:	FIM 4, 80H	
002A	517E		JMS TNFR	;Load RAM register from KBD
002C	2490		FIM 4, 90H	
002E	517E		JMS TNFR	
0030	24A0		FIM 4, 0AOH	
0032	517E		JMS TNFR	
0034	24B0		FIM 4, 0BOH	
0036	517E		JMS TNFR	
0038	4000		JUN START	;Input ended loop to START
003A	2420	LDOUT:	FIM 4, 20H	
003C	517E		JMS TNFR	

```
003C 2430          FIM 4, 30H
0040 517E          JMS TNFR
0042 2440          FIM 4, 40H
0044 517E          JMS TNFR
0046 2450          FIM 4, 50H
0048 517E          JMS TNFR
004A 2460          FIM 4, 60H
004C 517E          JMS TNFR
004E 2470          FIM 4, 70H
0050 517E          JMS TNFR
0052 2400          FIM 4, 00H
0054 5194          JMS TNFS        ;Load RAM register with '*'s
0056 2410          FIM 4, 10H
0058 5194          JMS TNFS
005A 5173  MSGO:   JMS RSET        ;End of calling message input
005C 21    WATB:   SRC 0           ;Wait for new KBD input
005D 115C          JCN 1, WATB
005F 00    SWB:    NOP             ;Short wait loop to insure
0060 715F          ISZ 1, SWB      ;KBD contacts closed
0062 2010          FIM 0, 10H
0064 21            SRC 0
0065 EA            RDR             ;Read ROM port 1
0066 BF            XCH 15          ;Load bit rate in IR(15)
0067 5173          JMS RSET
0069 21    WATA:   SRC 0           ;Wait for new KBD input then
006A 1169          JCN 1, WATA     ;transmit message
006C 2208  REPT:   FIM 2, 08H
006E 2600          FIM 6, 00H
0070 51A3          JMS LD8         ;Transmit 8 RAM registers
0072 5173          JMS RSET        ;one bit at a time
0074 2010          FIM 0, 10H
0076 21    WATC:   SRC 0           ;Wait for new KBD input
0077 1176          JCN 1, WATC
0079 00    SWC:    NOP             ;Short wait loop to insure
007A 7179          ISZ 1, SWC      ;KBD contacts closed
007C EA            RDR             ;Read ROM port 1
007D F4            CMA
007E 145A          JCN 4, MSGO     ;Retransmit message if ACC=0
008C 4000          JUN START       ;or loop to START
0082 2CCE  RECT:   FIM 12, 0CEH
0084 21            SRC 0
0085 F1            CLC
0086 EA            RDR             ;Read ROM port 0
0087 F5            RAL             ;Bit (3) into carry
0088 1A00          JCN 10, START   ;Loop to START if CY=0
008A 51F1  HTIM:   JMS ELV         ;High level pulse timing
008C 51F9  TIMH:   JMS SIXO        ;begins
008E 6F            INC 15          ;Keep track of number of
008F F1            CLC             ;timing cycles
0090 EA            RDR             ;Read ROM port 0
0091 F5            RAL             ;Bit (3) into carry
0092 128A          JCN 2, HTIM     ;Loop to HTIM if CY=1
0094 51F1  TIML:   JMS ELV         ;Low level timing begins
0096 51F9          JMS SIXO
```

```
0098 6E          INC 14          ;Keep track of number of
0099 F1          CLC             ;timing cycles
009A EA          RDR             ;Read ROM port 0
009B F5          RAL             ;Bit (3) into carry
009C 1A94        JCN 10, TIML    ;Loop to TIML if CY=0
009E F1    CKT:  CLC
009F AF          LD 15
00A0 9E          SUB 14
00A1 1AA5        JCN 10, ZERO    ;If IR(15)=IR(14) ACC will
00A3 14AC        JCN 4, CNT      ;equal zero and carry will
00A5 F1    ZERO: CLC             ;equal one.  If true loop
00A6 EA          RDR             ;to CNT, if not loop to ZERO
00A7 F5          RAL
00A8 12A5        JCN 2, ZERO     ;Loop to ZERO if CY=1 other-
00AA 4000        JUN START       ;wise loop to START
00AC 7DB2  CNT:  ISZ 13, TMH     ;Count four 10101 bit patterns
00AE 7CB6        ISZ 12, CK      ;in a row then jump to LSOH
00B0 4007        JUN LSOH
00B2 2E00  TMH:  FIM 14, 00H     ;Reset IR(14-15) to zero,
00B4 408C        JUN TIMH        ;loop back to TIMH
00B6 DC    CK:   LDM 0CH         ;Start wait out of four pulse
00B7 B8          XCH 10          ;time lengths
00B8 5225  KC:   JMS TMOT        ;Use basic time out cycle
00BA 00          NOP
00BB 00          NOP
00BC 7888        ISZ 10, KC
00BE DE          LDM 0EH
00BF BD          XCH 13          ;Reset IR(13) to 0EH
00C0 00          NOP
00C1 00          NOP
00C2 00          NOP
00C3 51F9        JMS SIX0        ;Timing cycle
00C5 408A        JUN HTIM        ;Four pulse timing over
00C7 2290  LSOH: FIM 2, 90H      ;loop to HTIM
00C9 21    SOH:  SRC 0
00CA EA          RDR             ;Read ROM port 0
00CB B9          XCH 9           ;Temporary store in IR(9)
00CC DC          LDM 0CH         ;Begin  time out cycle
00CD B8          XCH 8
00CE 00    TC:   NOP
00CF 78CE        ISZ 8, TC
00D1 00          NOP
00D2 5225        JMS TMOT
00D4 A9          LD 9            ;IR(9) back in ACC
00D5 1CC7        JCN 12, LSOH    ;Loop to LSOH if ACC not
00D7 72C9        ISZ 2, SOH      ;equal to zero, if equal to
                                 ;zero increment IR(2).  After
                                 ;seven consecutive zeros drop
                                 ;through
00D9 2200  REC:  FIM 2, 00H
00DB 2400        FIM 4, 00H
00DD 260A        FIM 6, 0AH
00DF 21    RECA: SRC 0
```

```
00E0 F1          CLC
00E1 EA          RDR            ;Read ROM port 0
00E2 F6          RAR            ;Three RAR's will put
00E3 F6          RAR            ;bit (3) into bit (0) of
00E4 F6          RAR            ;the ACC
00E5 B2          XCH 2          ;Save in IR(2)
00E6 521F        JMS RA         ;Timing cycle
00E8 5225        JMS TMOT       ;Timing cycle
00EA F1          CLC
00EB EA          RDR            ;Read ROM port 0
00EC F6          RAR            ;Bit (3) into bit (1)
00ED F6          RAR            ;
00EE 82          ADD 2          ;Add bit (0)
00EF B2          XCH 2          ;Save in IR(2)
00F0 521F        JMS RA
00F2 5225        JMS TMOT
00F4 F1          CLC
00F5 EA          RDR            ;Read ROM port 0
00F6 F6          RAR            ;Bit (3) into bit (2)
00F7 82          ADD 2          ;Add bits (0) and (1)
00F8 B2          XCH 2          ;Save in IR(2)
00F9 521F        JMS RA
00FB 00          NOP
00FC 5225        JMS TMOT
00FE F1          CLC
00FF EA          RDR            ;Read ROM port 0
0100 82          ADD 2          ;Add bits (0),(1) and (2)
0101 25          SRC 4
0102 E0          WRM            ;Write byte into RAM regis-
0103 65          INC 5          ;ter character
0104 5225        JMS TMOT
0106 7600        ISZ 6, RECB    ;Loop back to get another
0108 64          INC 4          ;byte,load same register
0109 7715        ISZ 7, RECC    ;Loop back to get another
                                ;byte, load new register
010B 411D        JUN VERF       ;Loop to VERF when eight
                                ;registers are filled
010D DD    RECB: LDM ODH        ;Begin delay timing
010E B8          XCH 8
010F 00    RD:   NOP
0110 780F        ISZ 8, RD
0112 00          NOP
0113 40DF        JUN RECA       ;End timing delay,loop RECA
0115 DE    RECC: LDM OEH        ;Begin delay timing
0116 B8          XCH 8
0117 00    RE:   NOP
0118 7817        ISZ 8, RE
011A 00          NOP
011B 40DF        JUN RECA       ;End timing delay, loop RECA
011D 2002  VERF: FIM 0, 02H
011F 2218        FIM 2, 18H
0121 2480        FIM 4, 80H
0123 520D        JMS CPAR       ;Compares two portions of RAM
                                ;to see if equal
```

```
0125 2220          FIM 2, 20H
0127 24A0          FIM 4, 0A0H
0129 520D          JMS CPAR
012B 2002          FIM 0, 02H
012D 2421          FIM 4, 21H
012F 2612          FIM 6, 12H
0131 5201          JMS LBLO        ;Subtracts one from first
                                   ;four bits of lable 1, stores
0133 A8            LD 8            ;results in IR(8)
0134 1447          JCN 4, SELT     ;IR(8) is equal to zero loop
                                   ;to SELT
0136 5201          JMS LBLO        ;Subtract two from lable 1
0138 A8            LD 8            ;store in IR(8)
0139 144D          JCN 4, ALL      ;IR(8)=0 loop to ALL
013B 5201          JMS LBLO        ;Subtract three from lable 1
013D A8            LD 8
013E 1453          JCN 4, GEO      ;IR(8)=0 loop to GEO
0140 5201          JMS LBLO        ;Subtract four from lable 1
0142 A8            LD 8
0143 1459          JCN 4, GRP      ;IR(8)=0 loop to GRP
0145 400           JUN START       ;Loop to START if lable 1 is
                                   ;in error
0147 2280  SELT:   FIM 2, 80H
0149 52CD          JMS CPAR
014B 415F          JUN DISPL       ;Displays lable 1 and mode
014D 2290  ALL:    FIM 2, 90H
014F 52CD          JMS CPAR
0151 415F          JUN DISPL
0153 22A0  GEO:    FIM 2, 0A0H
0155 520D          JMS CPAR
0157 415F          JUN DISPL
0159 22B0  GRP:    FIM 2, 0B0H
015B 520D          JMS CPAR
015D 415F          JUN DISPL
015F 2000  DISPL:  FIM 0, 00H
0161 2210          FIM 2, 10H
0163 5208          JMS DIS         ;Display mode
0165 2012          FIM 0, 12H
0167 2220          FIM 2, 20H
0169 5208          JMS DIS         ;Display lable 1
016B 5173          JMS RSET
016D 115F  ACK:    JCN 1, DISPL    ;If test pin equals one drop
016F 5173          JMS RSET        ;through, if not loop to DISPL
0171 4000          JUN START       ;Loop to START, wait for next
                                   ;message
           ;Subroutines RSET, TNFR, TNFS, LD8, DL, ELV,
           ;SIXO, LBLO, DIS, CPAR, RA, TMOT
0173 2810  RSET:   FIM 8, 10H      ;KBD reset subroutine
0175 29            SRC 8
0176 D8    SET:    LDM 8
0177 E1            WMP             ;Bit (3) of RAM port 1 is
                                   ;set to 1
0178 1976          JCN 9, SET      ;Loop to SET if test=1
```

```
017A 00          NOP
017B D0          LDM 0
017C E1          WMP            ;Bit (3) of RAM port 1 is
017D C0          BBL 0          ;set to 0
                                ;
017E 2680  TNFR: FIM 6, 80H     ;Transfer KBD to RAM routine
0180 5173  LDF:  JMS RSET
0182 21    WATD: SRC 0
0183 1182        JCN 1, WATD    ;Wait for KBD input
0185 00    SWD:  NOP            ;Short wait to insure KBD
0186 7185        ISZ 1, SWD     ;contacts closed
0188 EA          RDR            ;Read ROM port 1
0189 25          SRC 4
018A E0          WRM            ;ROM port 1 into RAM
018B 65          INC 5
018C 23          SRC 2
018D EA          RDR            ;Read ROM port 2
018E 25          SRC 4
018F E0          WRM            ;ROM port 2 into RAM
0190 65          INC 5
0191 768C        ISZ 6, LDF     ;Counts eight KBD characters
0193 C0          BBL 0
                                ;
0194 20A2  TNFS: FIM 0, 0A2H    ;Transfers '*'s into RAM
0196 2680        FIM 6, 80H
0198 A0    LDST: LD 0
0199 25          SRC 4
019A E0          WRM            ;First four bits of '*' into
019B 65          INC 5          ;RAM
019C A1          LD 1
019D 25          SRC 4
019E E0          WRM            ;Second four bits of '*'
019F 65          INC 5          ;into RAM
01A0 7698        ISZ 6, LDST    ;Counts eight '*'s
01A2 C0          BBL 0
                                ;
01A3 2408  LD8:  FIM 4, 08H     ;Load eight bits from RAM
01A5 27    LD8B: SRC 6          ;for serial transmission
01A6 E9          RDM
01A7 B0          XCH 0
01A8 67          INC 7
01A9 27          SRC 6
01AA E9          RDM
01AB B1          XCH 1
01AC 67          INC 7
01AD DE          LDM 0EH
01AE B4          XCH 4
01AF 23    LPS:  SRC 2
01B0 A0          LD 0
01B1 E1          WMP            ;Bit (0) transmitted
01B2 5225        JMS TMOT
01B4 51EB        JMS DL
01B6 00          NOP
```

```
01B7 00          NOP
01B8 23          SRC 2
01B9 A0          LD 0
01BA F6          RAR             ;Bit (1) into bit (0)
01BB E1          WMP             ;Bit (1) transmitted
01BC 5225        JMS TMOT
01BE 51EB        JMS DL
01C0 00          NOP
01C1 23          SRC 2
01C2 A0          LD 0
01C3 F6          RAR
01C4 F6          RAR             ;Bit (2) into Bit (0)
01C5 E1          WMP             ;Bit (2) transmitted
01C6 5225        JMS TMOT
01C8 51EB        JMS DL
01CA 23          SRC 2
01CB F6          RAR
01CD F6          RAR
01CE F6          RAR             ;Bit (3) into bit (0)
01CF E1          WMP             ;Bit (3) transmitted
01D0 5225        JMS TMOT
01D2 A1          LD 1
01D3 B0          XCH 0
01D4 74DD        ISZ 4, LPSA     ;Loop back and transmit
                                 ;second four bits of char-
                                 ;acter
01D6 75E6        ISZ 5, LD8A     ;Loop back get another
01D8 66          INC 6           ;character for transmission
01D9 73A3        ISZ 3, LD8      ;Loop back get new register
01DB 41EA        JUN RLD8        ;for transmission
01DD DD    LPSA: LDM ODH         ;Start timing delay
01DE B8          XCH 8
01DF 00    DLD:  NOP
01E0 78DF        ISZ 8, DLD
01E2 00          NOP
01E3 00          NOP
01E4 41AF        JUN LPS         ;End timimg delay
01E6 00    LD8A: NOP             ;Start timing delay
01E7             NOP
01E8 41A5        JUN LD8B        ;End timing delay
01EA C0    RLD8: BBL 0
                                 ;
01EB DC    DL:   LDM OCH         ;Timing delay subroutine
01EC B8          XCH 8
01ED 00    DLA:  NOP
01EE 78ED        ISZ 8, DLA
01F0 C0          BBL 0
                                 ;
01F1 DE    ELV:  LDM OEH         ;Timing delay subroutine
01F2 B8          XCH 8
01F3 00    LV:   NOP
01F4 78F3        ISZ 8, LV
01F6 00          NOP
```

```
01F7 00          NOP
01F8 00          NOP
01F8 C0          BBL 0
                                    ;
01F9 D3    SIXO: LDM 3              ;Timing delay subroutine
01FA B8          XCH 8
01FB 00    XO:   NOP
01FC 00          NOP
01FD 78FB        ISZ 8, XO
01FF 00          NOP
0200 C0          BBL 0
                                    ;
0201 F1    LBLO: CLC                ;Checks lable 1 character
0202 27          SRC 6
0203 E9          RDM
0204 95          SUB 5
0205 B8          XCH 8
0206 65          INC 5
0207 C0          BBL 0
                                    ;
0208 21    DIS:  SRC 0              ;Display routine
0209 E9          RDM
020A 23          SRC 2
020B E1          WMP
020C C0          BBL 0
                                    ;
020D F1    CPAR: CLA                ;Compares a series of RAM
020E 21          SRC 0            ;characters to see if they
020F E9          RDM              ;are the same
0210 23          SRC 2
0211 E8          SBM
0212 1C1C        JCN 12, ST       ;If ACC=0 and CY=1 both
0214 1A1C        JCN 10, ST       ;bytes the same, drop through
0216 61          INC 1            ;if not loop to START
0217 63          INC 3
0218 740D        ISZ 4, CPAR      ;Loop back and compare
021A 421E        JUN RBBL         ;anouther byte
021C 4000  ST:   JUN START
021E C0    RBBL: BBL 0
                                    ;
021F DC    RA:   LDM OCH            ;Timing delay subroutine
0220 B8          XCH 8
0221 00    RB:   NOP
0222 7821        ISZ 8, RB
0224 C0          BBL 0
                                    ;
0225 F1    TMOT: CLC                ;Primary timing delay based
0226 AF          LD 15            ;on contents of IR(15)
0227 F6          RAR
0228 1252        JCN 2, TW
022A F6          RAR
022B 1249        JCN 2, SIX
022D 1C3B        JCN 12, ONF
022F 4231        JUN SVN
```

```
0231 D7     SVN:   LDM 7
0232 B9            XCH 9
0233 D0     SVNA:  LDM 0
0234 B8            XCH 8
0235 00     SVNB:  NOP
0236 00            NOP
0237 7835          ISZ 8, SVNB
0239 7933          ISZ 9, SVNA
023B D9     ONF:   LDM 9
023C B9            XCH 9
023D D1     ONFA:  LDM 1
023E B8            XCH 8
023F 00     ONFB:  NOP
0240 00            NOP
0241 783F          ISZ 8, ONFB
0243 00            NOP
0244 793D          ISZ 9, ONFA
0246 00            NOP
0247 00            NOP
0248 00            NOP
0249 D2     SIX:   LDM 2
024A B8            XCH 8
024B 00     SIXA:  NOP
024C 00            NOP
024D 00            NOP
024E 784B          ISZ 8, SIXA
0250 00            NOP
0251 00            NOP
0252 D2     TW:    LDM 2
0253 B8            XCH 8
0254 00     TWA:   NOP
0255 7854          ISZ 8, TWA
0257 00            NOP
0258 00            NOP
0259 C0            BBL 0
END
```

BIBLIOGRAPHY

US XI/3(c)/30, 29 January 1973, 11th session of the Sub-Committee on Radiocommunications for the CCIR.

COM IX/2(c)/4, 30 December 1971, ninth session of the Sub-Committee on Radiocommunications for the CCIR..

COM X/3(c)/3, 16 June 1972, tenth session of the Sub-Committee on Radiocommunications for the CCIR.

MCS-4 Micro Computer Set, USERS MANUAL, March 1972, Rev. 2. INTEL CORP. 3065 Bowers Ave, Santa Clara, Calif.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defence Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Professor V. M. Powers, Code 52 Pw Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	1
4.	LT Vernon C. Hipkiss, USCG (student) 800 Agua Caliente Rd. Sonoma, California 95476	1

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Selective Calling Device for The Mobile Maritime Service		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis;(September, 1973) 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Vernon Christopher Hipkiss		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		12. REPORT DATE September 1973 13. NUMBER OF PAGES 43
14. MONITORING AGENCY NAME & ADDRESS(If different from Controlling Office) Naval Postgraduate School Monterey, California 93940		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release;distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

SELCAL, Digital calling device, Selective Calling Device, MCS-4 applications.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

INTEL's MCS-4 micro computer set was selected to implement a digital selective calling device for use by the mobile maritime service. The system will generate and receive a standardized calling message. Upon receipt of a valid calling message, intended for the user, the system will activate visual and audible alarms.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601 |
(Page 1)

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

23 MAR 76
7 JUN 77
S 9364
24157

Thesis
H5745 Hipkiss
c.1
146101
Selective calling device for the mobile maritime service.